



EMUL166 E3-PC

Getting Started Manual

Version 1.1

Contents

Chapter 1 The Nohau EMUL166: The Hardware Parts	4
Some Background: the Nohau EMUL166-PC Emulator	4
The History of the EMUL166 Emulator:	4
The Two-board EMUL166 Emulator Design:	6
Emulator Pod Jumpers, LEDs, and Test Points:	9
Emulator LED's:	9
Emulator Pin-outs (in numerical order)	10
Edge Jumpers (in physical order)	11
Default Jumper Settings of the E3 Emulator POD -167N Rev E, F and G	12
Trace User Input Connector: (J3)	13
Chapter 2 The Nohau EMUL166: The Software Parts	14
The Nohau Universal User Interface Seehau	14
Configuring the Emulator Software Seehau.	14
Important Software and Hardware Notes:.....	16
Starting the Emulator and Seehau	16
Problems ?	16
Shutting down Seehau:	17
Seehau Commands	18
Macro Construction	18
Macro Execution	18
Command Format.....	19
Command Examples	19
Command Groups:	20
Quick Saving a Configuration Menu using the Apply Button	20
Creating New Buttons	21
Chapter 3 The Nohau EMUL166: Timer.abs Example	22
Running the example program TIMER.ABS	22
Demonstrations of Seehau Features	23
Watching Data in Real-time with the Shadow RAM	23
Graphical Display of Data in Real-Time Using a Gauge	24
Graphical Display of Data in Real-Time Using a Graph	25
Setting Breakpoints	25
Program Performance Analysis (PPA)	26
Chapter 4 The Nohau EMUL166: Trace and Triggers	28
Trace and Trigger Overview	28
Trace Window Display.....	28
Trigger Example on an Address and Data Qualifier.....	29
Trace Filter Example	31

Chapter 5 The Nohau EMUL166: Sabre Trace and Triggers 32
Trace and Trigger Overview 32
 Trace Window Display 32
 Trigger Example on an Address and Data Qualifier 33
 Trace Filter Example 35

Chapter 6 The Nohau EMUL166: Connecting to your Target 36
Clocks, Oscillators and Crystals 36
 Crystal Capacitors 36
 EMUL166 Oscillators 36
Connecting to Your Target Hardware 37
Conclusions 37

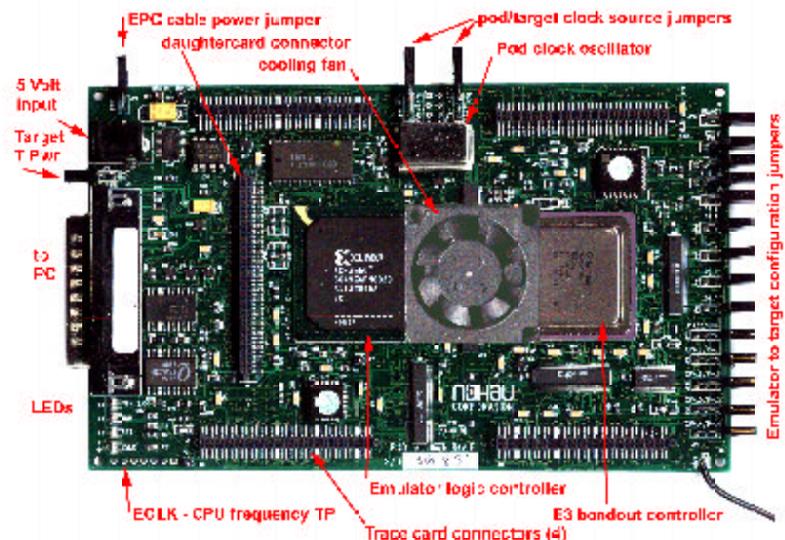
Chapter 1 The Nohau EMUL166: The Hardware Parts

Some Background: the Nohau EMUL166-PC Emulator

The History of the EMUL166 Emulator:

The 3 Nohau emulator generations: E1, E2 and E3.	There are currently three generations of the EMUL166 emulator that supports the Infineon C166 family of microcontrollers. The Nohau EMUL166 uses one of three versions of an Infineon bondout microcontroller (E1, E2 or E3) as the emulation processor. This special processor contains a C166 core surrounded with special emulation logic that provides emulator access to the C166 internal busses. This allows the emulator visibility into the C166 core and triggers on internal reads, writes and ALU results.
Internal (single-chip) and external modes	The bondout supports both internal (single-chip) and external memory modes. Special emulation RAM in the emulator provides 32 bit ROM emulation with the bondout 32 bit ROM bus. This 32 bit RAM provides full speed program fetch and execution. See the "The Embedded Software Engineer's Guide to In-Circuit Emulation" for more detailed information. This document is available from www.icetech.com or any Nohau representative or distributor.
E1 Bondout emulator: the first generation.	The first generation EMUL166 consisted of a two board design for the emulator and two trace boards in a sandwich design. One trace board was for internal memory and the other external memory. The bottom emulator board contained the Siemens E (E1) bondout controller and C167CR and C163 controllers that were used to provide the CAN and SSP XBUS peripherals. This emulator operated at a maximum speed of 25 MHz.
E2 bondout: 33 MHz with one board plus a trace board.	The second generation uses the Infineon 33 MHz E2 bondout which contains two CAN controllers. The four boards of the E1 emulator have been redesigned into two: one emulator and the optional trace board. The Infineon two chip emulation technique is used to provide XBUS peripherals that the bondout does not. J1850 and I ² C are two examples. A small daughtercard is provided that supplies these XBUS peripherals. The emulator board contains the emulator logic including a FPGA that provides great flexibility in updating the firmware. This emulator is commonly known as the "E2" and is still a useful debugging tool.
E3 bondout: 40 MHz plus a new trace board option.	The third generation of the EMUL166 uses the E3 bondout device and its speed increases to 40 MHz. Functionally it is similar to the E2 emulator. A new trace board was introduced in 2001 to enhance triggering and tracing capabilities. This emulator is known as the "E3" and was introduced late 2000. Figure 1 shows the EMUL166 emulator connected to the Keil C167CR evaluation board.

Figure 1
The EMUL166 connected to a target board



Seehou: the debugger software	The Nohau debugging software is called "Seehou" and the emulator and its software is designed to be relatively intuitive to use. Seehou updates are available free on the website or directly from any Nohau office or rep anywhere in the world. Seehou is macro based enabling automatic operation. Seehou operates under Windows 95, 98, NT, ME and 2000.
V2 and V3 trace boards	The trace board designed for the E2 emulator is called the "V2". The V2 will operate with the E2 or E3 emulators. The V3 board was introduced in 2001 and will operate only with the E3 emulator and has many advanced features including expanded abilities to trigger on internal reads. The trace board allows tracing on both internal and external memory areas and is optional but can be added later.
Triggers	The trace board contains a memory buffer and trigger logic. You can set triggers on specified addresses and data which will stop the emulation and/or trace memory when this action occurs. This alerts you that the specified event has occurred and you may now use the information stored by the emulator to find any hardware or software errors. The trace board contains trigger in and trigger out connectors plus 8 bit data inputs that are recorded in the trace memory.
Trace memory	The trace memory records the microcontroller cycles including data reads and writes for user specified conditions. You can view the trace memory to find out what your code was actually doing at a particular time. Most people purchase the optional trace card due to its unique ability to save many hours of engineering time looking for elusive bugs.
Connecting the PC and to the target	The EPC cable is shown in Figure 1. The Emulator Parallel Cable (EPC) connects to the LPTx port of the host PC. An ISA card (LC-ISA) and a USB cable is also available. Various adapters are available to connect to virtually any target system. These options are all listed with photos in the Nohau C166 Price List.
Emulating new XBUS peripherals	The bondout will not incorporate all the C166 family peripherals of a particular derivative to be supported. Examples are the I2C, J1850 ports and extra XRAM. These are all situated on the XBUS which is an external representation of the external bus. The problem of providing these peripherals is solved with the Infineon two chip emulation system.
The XBUS daughtercards	A daughtercard containing a production chip which will provide the XBUS peripherals to be emulated is plugged into the emulator. The emulator firmware then places this chip into emulation mode. The bondout can now access these peripherals as if they were in the bondout itself and still at full speed.
Do I need a daughtercard?	A major advantage is you are given the latest peripheral modules from the daughtercard chip. For this reason, Nohau will use all the modules from the daughtercard even if these modules are in the bondout. Check the C167E3 Product Support Table in the price list to see if you need to install a daughtercard to support your target microcontroller.
The C166, ST10, C166S-V2 and the Super10 families	Infineon and STMicroelectronics provide the C166 and ST10 families respectively. These families are related and compatible to some extent. Nohau makes three emulators for the Infineon C166 and C166S-V2 families and three for the ST10 and Super10 families. It is possible to provide support for Infineon chips with the ST10 emulator and vice versa. Contact Nohau technical support for cross support of various devices and their specifications.
The ST10 emulators	The EMUL-ST10 emulators provide some Infineon support to speeds of 50, 90 and 100 MHz. This emulator has advanced trace and trigger facilities for power users. Nohau does not support the SAB-80C166 part itself. All other derivatives are fully supported by the appropriate emulator(s). Call Nohau or your local rep for the latest support list.
OCDS Support	Nohau makes a OCDS (on chip debugging support) debugger for the Infineon series of parts having this module. Currently these are the C161U and the C165 UTAH and the C165H. This debugger uses the JTAG port on these devices. The Nohau OCDS debugger provides Level One support today. Level Two includes trace and trigger features and is not implemented yet.

The Two-board EMUL166 Emulator Design:

The EMUL166 E3 emulator consists of an emulator pod board with an optional trace board. The emulator board contains the Siemens E3 bondout controller and various logic in the form of advanced CPLDs. This board will operate as an emulator but without the trace, trigger or shadow RAM features. An optional second board, the trace memory, plugs on top of the emulator pod board and provides these features. The current maximum speed of both boards is 40 MHz. The photo on the front cover of this document shows the complete emulator in its case and connected to a PC LPT1: port with the EPC cable. The emulator is configured and operated by the Nohau user interface, Seehau. Seehau also contains the firmware for the CPLDs of the emulator. Firmware updates are easy and complete with each new release of Seehau. There are no separate firmware loaders necessary for Nohau products.

The emulator board has various jumpers, LEDs and test points to configure the emulator hardware to your specifications. Figure 2 is a photo of the top of the emulator board and shows some of the user jumpers, connectors, other features and the E3 bondout controller. Figure 3 shows the emulator board in a line drawing. Figures 4 and 5 show the trace board. The trace board does not have any user jumpers available as they are all reflected on the emulator board or are software configured by Seehau. All trace options are software configured by the Seehau interface.

On the bottom of the emulator pod are 5 connectors in an industry standard configuration designed to connect the bondout emulation controller to the target hardware. This connection can be direct with the appropriate connectors designed on the target board, or through various adapters available from Nohau. See the C166 price list for details on target adapters. Figure 6 is a photo of the bottom of the emulation board with the bottom case installed. Note the 4 large connectors (J5 to J8) and a single small one (JP46). JP46 routes XBUS peripheral signals from future derivatives to be sent to the target. Maximum system speeds are maintained by not having these XBUS signals switched by logic. These signals will normally be routed physically by the adapter or the target board.

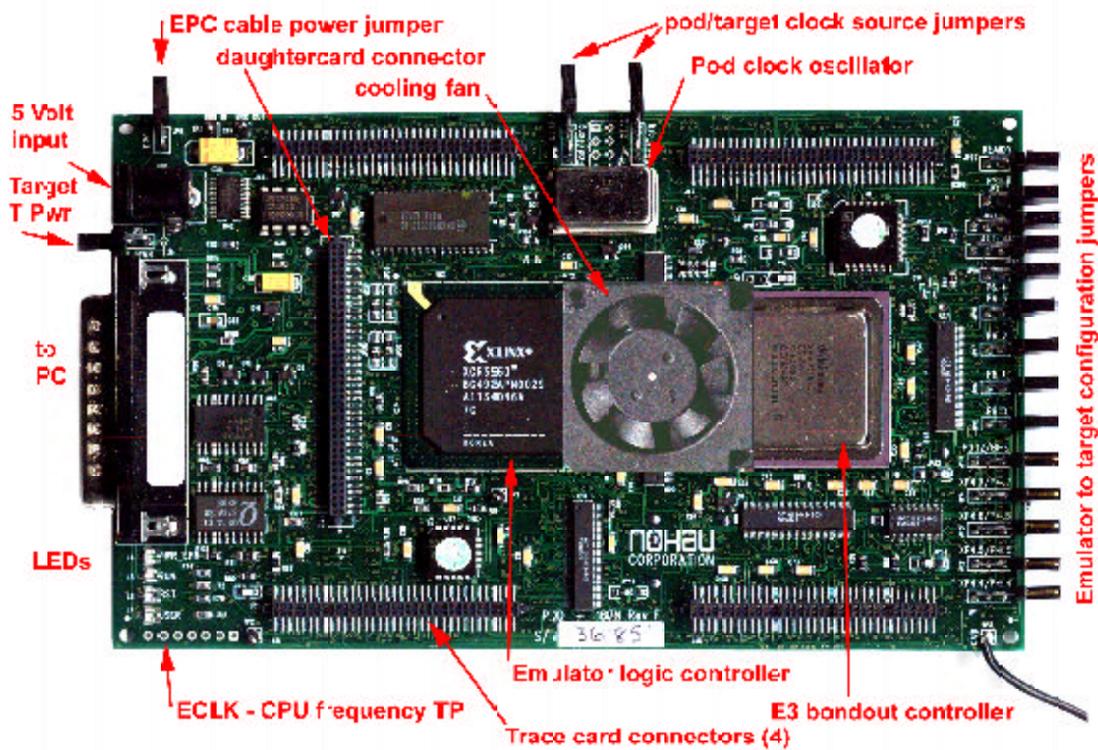


Figure 2
Emulator Board

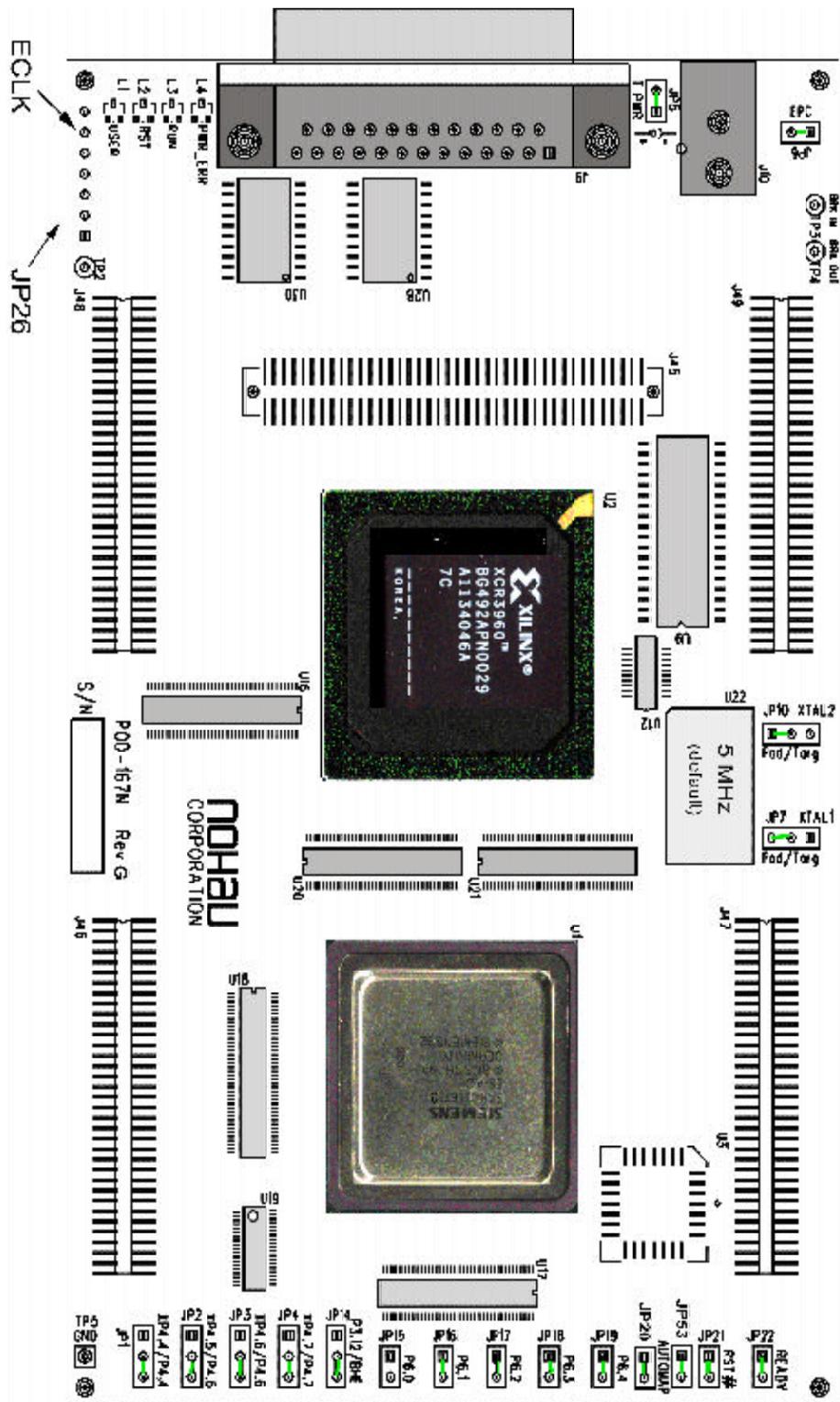


Figure 3
Emulator Board

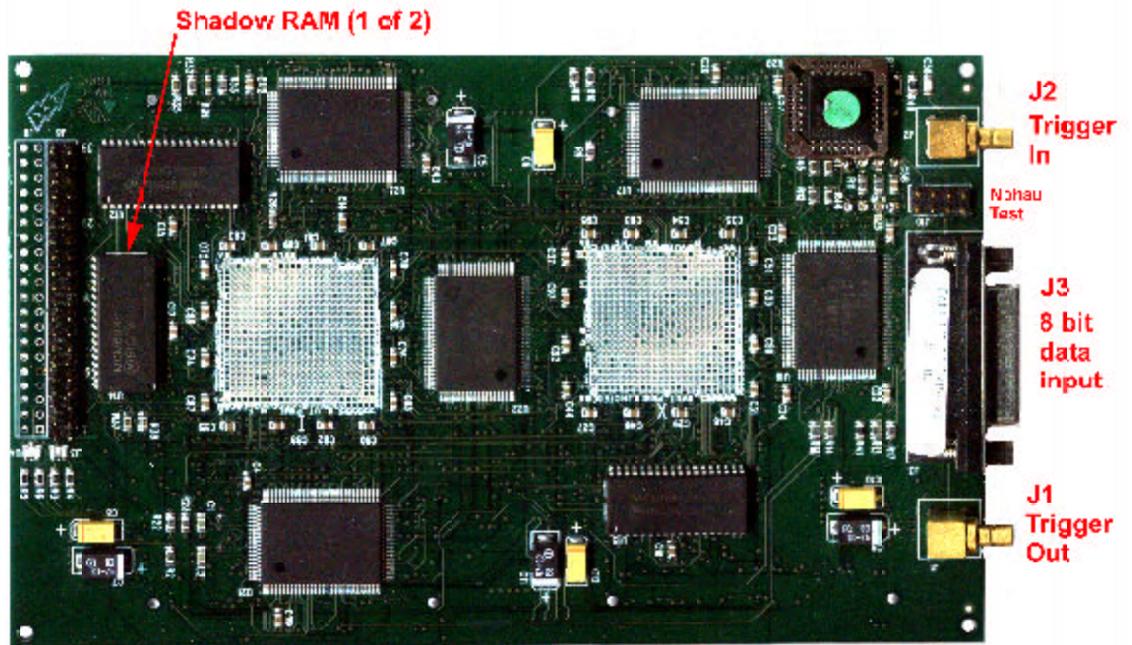


Figure 4
Trace Board Top

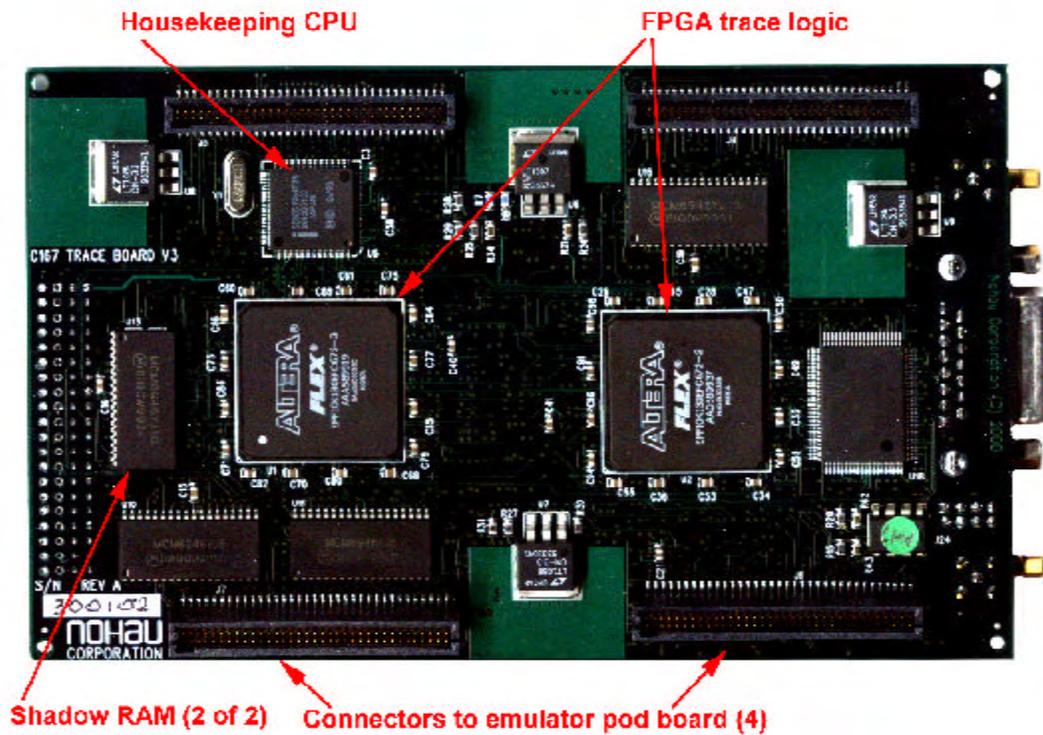


Figure 5
Trace Board Bottom

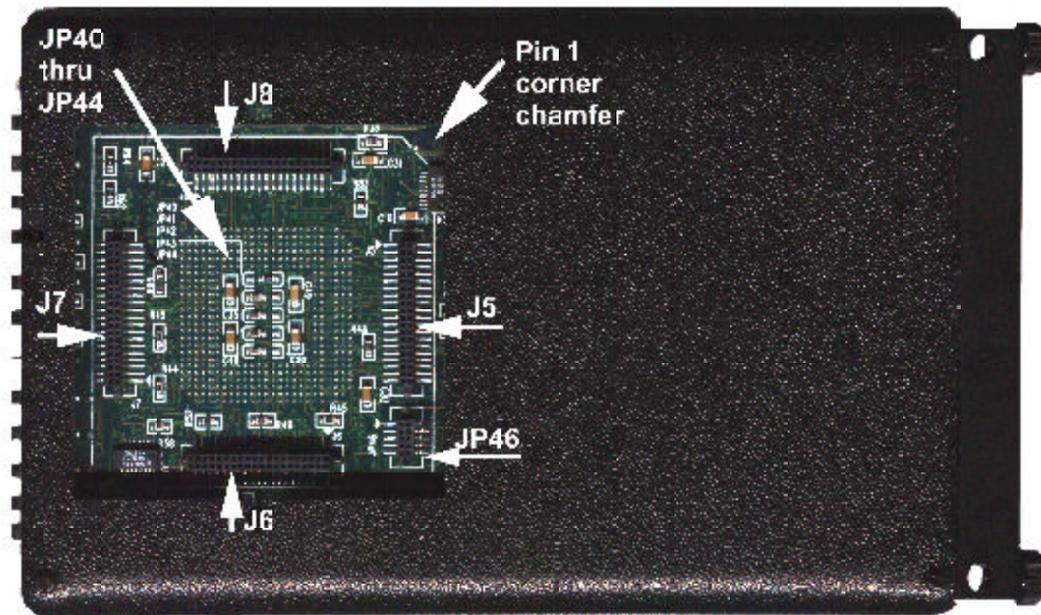


Figure 6
Target Connectors

Emulator Pod Jumpers, LEDs, and Test Points:

This section is based on the E3 POD167-N Revision F or G. This information is clearly marked on the upper side of the emulator board between the trace board connectors J48 and J46. This is the edge opposite the two clock jumpers. This is shown just above the serial number location in Figure 3. Figure 2 and 3 show the location of the various jumpers and test points. Jumper information is also available in the Seehau on-line help. There are no user jumpers for the trace board.

Emulator LED's:

Normal operation with emulation stopped has all LEDs L1 to L4 off. With a user program running, L3 (RUN) will go on. The red RESET LED L2 lights when Seehau is performing a system reset.

L1 USER

This is a yellow LED the user can affect by applying standard TTL levels to TP2. TP2 is active low - grounding TP2 turns on L1.

L2 RST - RESET

This red LED indicates the application of a reset signal by the Seehau software. The emulator will be in the reset state when powered with 5 volt supply but without a cable connected to the PC connector J9.

You can reset the emulator by selecting Reset Chip in the Run menu or clicking on the RESET icon on the toolbar. The commands *run_reset* and *run_fullreset* will also reset the emulator. These commands can be entered in the command dialog box or through a Macro.

L3 RUN

This green LED indicates whether the Nohau monitor program or a User program is running. L3 illuminated means the user program is running.

L4 PWR_ERR - Power Error.

If this LED lights, the correct 5 volts supply is not connected. This can occur if the 5 volt regulated supply is not connected to J10 and the PC communications connector J9 is connected. L4 is a red LED.

Emulator Pin-outs (in numerical order)

TP1: Not present.

TP2:

This input activates LED L1 USER. TP2 is active low: this action turns L1 on.

TP3: Break In

This input will cause a break in program execution. TP3 is usually connected to TP4 Break Out from another EMUL166 emulator. This feature is used to synchronize more than one emulator in a multi-processor system. Contact Nohau technical support at support@icetech.com for more information.

TP4: Breakout:

This output indicates if the emulator is in monitor mode or is running user code. It is connected to the RUN LED L3. This output is usually used for customers who have multiple emulators connected to the target board. It provides a means to synchronize them together. TP4 is high when the Nohau monitor is running.

TP5: GND

The emulator ground connection and a black ground wire with a clip is soldered to this point. Connect this clip to a secure ground (or earth) connection on your target hardware. Do this before connecting the emulator processor pins to the target.

TP6:

This is for Nohau testing. There is no user use for this test point.

JP7 and JP10:

XTAL1 and XTAL2 - These jumpers determine whether the on-board pod clock (U22) or the target clock is used for emulation. These jumpers must be changed in tandem as a pair. The default is set to POD (the upper positions) and U22 provides the clock signal to the POD controller. The Pod will supply the clock to the target if you jumper all three of the pins together. In this case, make sure the target clock is disconnected.

JP5: T PWR

+5 volts is supplied from the pod power supply through JP5 to the bondout controller. During normal operation with a target board, 5 volt power for the bondout controller U1 will be supplied by the target and JP5 must be removed. The pod power supply must still be connected in order to supply power for the rest of the emulator.

JP5 is used for standalone operation i.e. without a target board. JP5 will send power to the target if left jumpered. If the target has power, this can cause a conflict so JP5 must be removed. If the target has no internal power supply, the pod is able to supply a small amount, less than 500 ma, to power the target. This may be useful for very small targets with low current consumption.

Note: When powering up or down the system: the target must never be powered when the pod is not. Power flowing from the target into the bondout controller will probably damage it. Always power up the pod first, and power it down last. Do not leave the Pod powered up without the target power applied for more than 2 minutes. This situation stresses the I/O circuitry of the bondout chip and will cause permanent damage. Do not leave the pod powered without the Seehau software running it. The pod will be in an uninitialized state with unpredictable results. There are no reports of this causing damage.

Power the emulator by switching the AC mains power on after connecting the 5 volt plug to J10. Turn-on transients caused by the 5 volt plug bouncing may lock up the bondout and CPLD chips. It is convenient to switch AC power on and off using a power bar rather than by inserting the DC plug into J10.

JP26:

This unpopulated connector is used for testing the pod by Nohau. Pin 6 (ECLK) can be used to measure the CPU clock frequency. The C166 family has a clock prescaler and a PLL on chip providing scaling of oscillator U22. This pin is the second from the end of the Pod board closest to L1, the USER LED.

Pin 7 (EMUL#) indicates if the Pod is in USER or MONITOR mode. This pin is closest to the edge of the board. This signal is connected to the RUN LED. The rest of the pins provide no useful information.

JP6:

EPC PWR - JP6 supplies the EPC cable (Emulator Parallel Cable) with 5 volts. If this special cable is not used, remove JP6. The LC-ISA board does not need power from the pod and JP6 must be removed.

JP13:

This 8 pin connector is used by Nohau to program the serial EEPROM. It has no user use.

J45:

This connector will connect to a small daughterboard to supply XBUS peripherals not supported by the bondout controller. This allows the emulation of derivatives not yet introduced and prevents obsolescence of the emulator. Some of these signals will be sent to the target through JP46 on the underside of the Pod. See Figure 6. JP1 through JP4 control the path of signals from J45 or from the bondout controller U1.

J46, J47, J48 and J49:

The optional Trace memory board connects to these four connectors.

J5, J6, J7 and J8:

Connects the bondout controller signals to the target. This is an industry standard connector and can be directly connected to appropriate target boards or through various adapters available from Nohau.

JP46:

This 10 pin connector is used to send XBUS peripheral signals from the optional daughterboard to the target. This connector and J5 through J8 are shown in Figure 6.

JP40 through JP44: (see Figure 6)

These five jumpers have zero ohm resistors soldered in them. These jumpers direct logic flow for the five C166 chip selects. Normally, you do not need to change these jumpers. Contact Technical Support.

Edge Jumpers (in physical order)**JP22: READY:**

JP22 connects the READY input signal between the target and the POD controller. The READY signal is used to terminate a bus cycle and is useful when external devices have a long or indeterminate access times. READY is activated in certain address ranges as determined by the BUSCON registers.

If the READY signal is not asserted for a bus cycle that it is activated for in BUSCON, the processor will stop. Only a reset or a watchdog timeout can restore operation at this point (or the assertion of READY). Since the bondout controller on the pod is used for some housekeeping when emulation is not running, suspending it can cause Seehau to stop responding. Remove JP2 to prevent this. The default is connected.

JP21: RST#

JP21 connects the Pod and target reset pins together. The RST is the master reset pin. A target peripheral with the ability to assert this signal may not be desirable. Remove the jumper on JP21 to prevent the Pod controller from being reset from the target. A reset on the Pod controller will not be passed to the target board if this jumper is not connected. JP21 is connected in by default.

JP20 and JP53: AUTOMAP:

Determines whether memory mapping is controlled by software with Seehau or the chip select pins setup contained in the ADDRSEL1 through ADDRSEL4 registers. The Map to Target Memory window is the software mapping system. It is found by selecting Config, Emulator in the main Seehau window and then selecting the MEM Map Config tab. AUTOMAP determines whether the numerical addresses specified in the window or the chip selects selected are passed to the target board. If JP20 and JP53 is connected, then software mapping is activated. Since there are no entries in the mapping window, all memory is mapped to the emulator. If JP20 and JP53 is removed, then mapping is determined by the chip select jumpers and the BUSCON and ADDRSEL registers. The default is JP20 and JP53 connected in the upper position. These two jumpers must be moved or installed in tandem. If one is up, the other also must be.

JP15 through JP19:

The controller provides 5 programmable chip select pins (CS0-CS4) that are output on parallel port 6 if it is programmed as an Alternate Function. These are pins P6.0 through P6.4. Recall that if these pins are not used as chip selects, they can be used as general purpose I/O ports.

JP15 through JP19 connect these 5 pins to the target. They create the mapping signals from the chip selects as in previous Nohau C166 emulators. The default is JP16 through JP19 connected. JP15 not connected.

JP14: P3.12 or BHE:

If you configure P3.12 as an I/O pin, place JP14 on the P3.12 side (upper position). If you configure P3.12 as either BHE or WRH, place JP14 on the BHE side (lower position). The default is BHE (lower position).

In the process of mapping sections of memory to the target, the RD, WR and BHE signals are not passed to the target directly. The mapping function of the emulator switches off the pod memories as required. However, as the signals map memory access to the pod, it gates off the RD, WR and BHE signals from the bondout controller to the target.

When P3.12 is configured as a standard I/O pin, you would not want it gated off by the pod logic. When the jumper shunt is in the P3.12 position, it passes directly from the bondout controller to the target. When the jumper shunt is in the BHE position, P3.12 gates off and pulls high during memory accesses to the pod as required. The default is BHE connected (or the lower position).

JP1 through JP4:

XP4.4/P4.4 through XP4.7 through P4.7: These jumpers select whether the upper 4 bits of Port 4 supplied to the target come from the bondout controller or from a daughterboard connected to J45.

The XBUS is an internal representation of the external bus. This bus is used internally to connect various peripherals to the CPU. These peripherals look like external devices to the CPU even though they are on-chip and are accessed as such. Internal peripherals are also on the XBUS.

The bondout controller U1 used for emulation contains most peripherals. If a certain Xbus Peripheral is available on the bondout chip on the pod, then this peripheral will be used to provide the peripheral to the target. For those derivatives that use peripherals not on the bondout controller, a daughterboard connected to J45 will contain a regular series controller that contains this XBUS peripheral. JP1 through JP4 must be set so these signals are sent from the daughterboard to the target. (XP4.4 through XP4.7 - the upper positions) The default is all jumpers are set to P4.x (the lower positions).

Default Jumper Settings of the E3 Emulator POD -167N Rev E, F and G

This list is for stand alone operation without a target connected. Do not change any trace board jumpers.

POD board

JP7 and JP10 upper position (Pod). This selects the emulator clock and not the target clock.

JP5 on. This is T PWR and supplies 5 volts to the bondout controller.

JP6 off (on if using the EPC cable)

JP20 and JP53 upper position. This is the AUTOMAP control jumpers. This selects all emulation RAM.

The 13 jumpers on the edge of pod board are all connected except for JP15 (P6.0). If a jumper has 3 pins (JP16 to JP20), the position is lower or close to the circuit board. JP20 and JP53 are in the upper position.

Trace User Input Connector: (J3)

The trace board contains a 15 pin D shell connector. Refer to Figure 4 for its location. This connector provides 8 user input signals to be recorded in the trace memory under the Misc. column. These are TTL level inputs. The Trigger In (J2) is also available on this connector. A special socket with clips is available from Nohau as part number TR167 Probe. The pin assignments are as follows:

Name	Pin	Lead Colour of Trace Probe Cable Assembly
External In 0	2	black
External In 1	3	brown
External In 2	4	red
External In 3	5	orange
External In 4	6	yellow
External In 5	7	green
External In 6	8	blue
External In 7	9	violet
Vcc 5 volt	1	not connected in Trace Probe cable assembly
Trigger In	11	red clip -white wire
Trigger Out	13	green clip -white wire
Ground	15	grey

Chapter 2 The Nohau EMUL166: The Software Parts

The Nohau Universal User Interface Seehau

The Seehau Macro based GUI is designed to provide a consistent user friendly interface for all Nohau in-circuit emulator families. Seehau is a High Level Language (HLL) debugger that allows you to load, run, single-step and stop programs, set and view trace and triggers, modify and view memory contents including SFRs, and set software and hardware breakpoints. Seehau runs under Windows 95, 98, Me, NT and 2000.

Seehau has the capability to run over a TCP/IP stack. Seehau is also an OLE Automation server. This means that Seehau based emulators can be manipulated from an application developed in any environment that supports OLE Automation. OLE (Object Linking and Embedding) Automation allows one application to drive another application. The driving application is known as an automation client or automation controller, and the application being driven is known as an automation server or automation component. You can control Seehau from C++, Java, Delphi or Visual Basic.

Configuring the Emulator Software Seehau.

The commands and data used to configure Seehau when it is started is contained in the file startup.bas. The file startup.bas is an ASCII file in the default directory c:\nohau\Seehau16x\Macro. This file is created by the Seehau Configuration program using user supplied information about the emulator and its environment. It is not created when Seehau is initially installed. The file seehau.ini in the c:\nohau\Seehau16x directory specifies this startup file.

The configuration can be started by clicking on the Seehau Config icon on your desktop. If you start Seehau itself and startup.bas does not exist, Seehau will start the configuration program. This is a good method to totally reconfigure the emulator software. Simply delete or rename startup.bas and you can create a new one.

Note you do not need to have the emulator connected to the PC to run the Seehau Config program. You do need the emulator connected and with the jumpers properly set in for Seehau to operate properly. For more information on macros; see the Macro Section in this manual.

You can access the emulator and trace setup windows from within Seehau under the Config menu item in the main window. Click on Emulator or Trace for the appropriate setup desired. Note that more detailed setup options are available this way.

This manual assumes you are able to load the Seehau software. Make sure this is done now. If you have trouble loading from diskettes, copy them to a temporary directory on your hard disk and install from there.

It is better to get familiar with the emulator in stand-alone mode before attempting to connect to a target hardware system. The added complications of the target hardware may cause you undue problems at this time. Once you have gained some skills at operating the emulator, it will be easier to connect to your target.

- 1) Click on the Seehau Config icon on your desktop. You do not need the emulator connected at this time.
- 2) A blank Figure 1 will open. You will now configure the emulator. You will need to know what interface connector you are using: EPC or the LC-ISA card.
- 3) Change the settings as indicated. Make sure you only select an E1 selection if you have the E1 bondout controller on the original emulator series. The EPC cable and the LC ISA card are the only appropriate choices for the EMUL166. Figure 1 shows the settings used if you have are using the EPC cable. The EPC cable is distinguished by a male/female connector on the end that plugs into your PC parallel port. Figure 2 shows the settings for the LC-ISA card. The Emulator Board Address dialog box is for the address of the internal communication link from your computer. For the ISA card, the most common address is 200. This setting can be changed on the board. If you are using the EPC (Enhanced Parallel Cable), this address is not applicable. The EPC cable normally uses address 378 which represents the LPT1 port on your PC. It can use other LPT addresses as well.

When all the information has been entered in Figure 1 or 2, pressing Next will open up Figure 3. It is possible to operate the emulator remotely via a TCP/IP stack. Contact Nohau Technical Support at support@icetech.com

Important Software and Hardware Notes:

Always use Uninstall to unload any existing version of Seehau from your computer before loading in another copy. Do not simply delete the Seehau files and/or folders. Do not install Seehau on top of an existing copy. Seehau will not allow this action. Under My Computer select Control panel, then Add/Remove Programs. After Uninstall has run, you might want to delete any files and folders that Uninstall did not delete. Save your personal macros and source files first. Pay particular attention to startup.bas in the Macro directory. You may not want to use your old copy of this file so make sure it is erased.

Do not leave the emulator powered for extended periods of time without Seehau active. The pod will be in an uninitialized state with unpredictable results. There are no reports of damage resulting from this.

Pay attention to the power-up and power-down sequences of the emulator and a target system. When powering up or down the system: the target must never be powered when the pod is not. Power flowing from the target into the bondout controller will damage it. Power up the pod first, and power it down last.

Starting the Emulator and Seehau

- 1) The two clock jumpers JP7 and JP10 must be in the upper position i.e. "POD".
- 2) T Pwr (JP5) must be connected. This jumper supplies power to the bondout CPU. This jumper is located between the PC connector and the power supply socket.
- 3) If you are using the EPC, the EPC power jumper (JP6) must be connected to send power to the EPC. This jumper is located beside the power socket opposite side of the T Pwr jumper. If you are using the ISA card, this jumper must be left open or the power from the ISA will conflict with your power supply.
- 4) All the 13 jumpers on the end edge of the board must be connected except for JP15 (P6.0 or CS0). For the 5 that have dual pins, the position is lower or close to the PC board. All these jumpers may not absolutely be needed in order for this example to work.
- 5) Connect the cable between the PC and the 25 pin connector on the emulator board.
- 6) It might be a good idea to double check your settings.
- 7) Plug in the Nohau 5 volt power supply.
- 8) The green RUN and perhaps the red RST LEDs will illuminate with the EPC cable connected.
- 9) Double click on the Seehau 16x icon on your PC desktop.
- 10) Seehau will configure itself from startup.bas and may present this message asking if you want to reset the emulator. The green RUN LED is probably on: Click on Yes. This gives you the opportunity to reconnect to a running emulator if you had shut down Seehau previously,. "Mac" will be displayed in red at the bottom of the main window while startup.bas is running.
- 11) The red Reset LED will then light and then both it and the Run led will go out. The Seehau interface will finish loading itself. Position and size the main window to your preference. You can open up new windows. They are found in the New menu item on the main Seehau window.

Problems ?

Problems are usually from the EPC PWR, T PWR or the clock incorrectly connected. Review instructions 1, 2 and 3. Make sure the cable to the PC is correctly connected. If are using the EPC, try it without a printer connected. Do not have the emulator connected to a target system or adapter. You should have a 5 MHz oscillator module installed to get a 20 MHz CPU frequency. Set your PC parallel port for "Output Only" in the BIOS setup. It may not work well in ECP or other modes. Turn on the 5 volt supply by connecting the 5 volt cable to J10 first, then connect the mains AC power. This creates fewer turn-on transients.

Try another PC. Reload Seehau. If you do this, use the Windows Remove Programs found under My Computer, System. This way, all files and registry entries are properly deleted.

If all this fails, please contact your Nohau representative for technical support. support@icetech.com.

I set my windows up as in Figure 4. I resized the existing windows and opened up the Trace window.

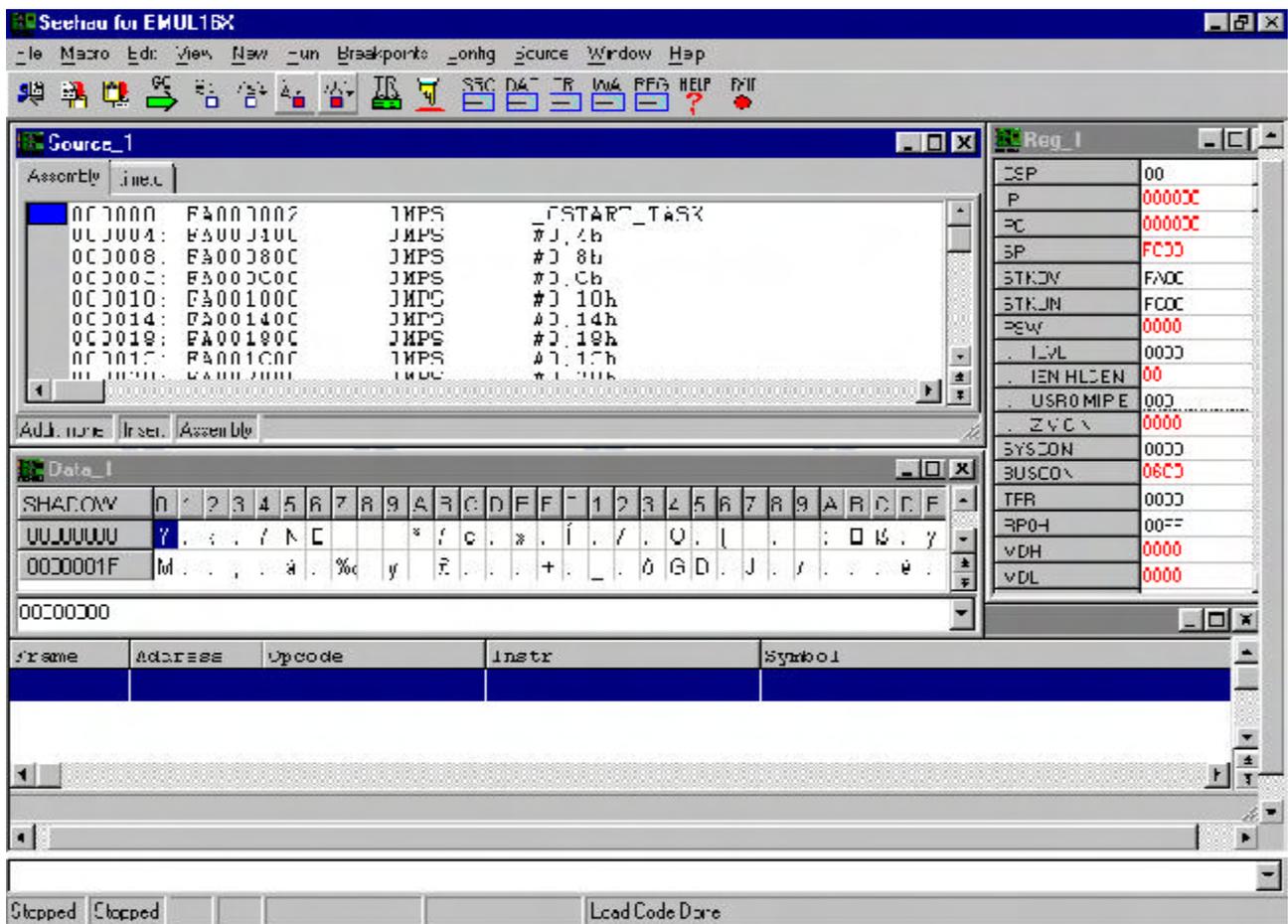
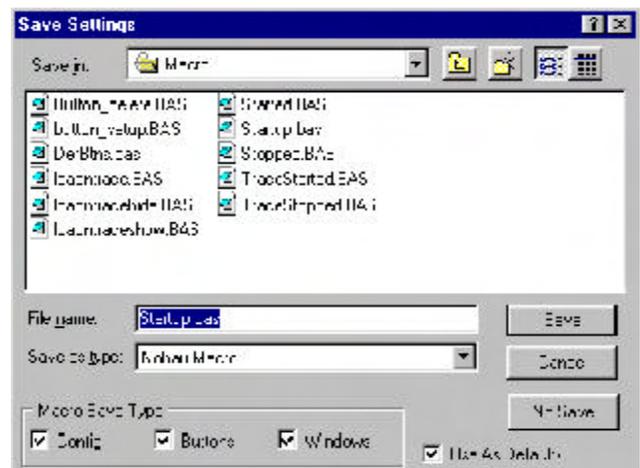


Figure 4

Shutting down Seehau:

- 1) Click on the X in the upper right corner or select the menu item File, Exit. Figure 5 will appear.
- 2) You can save your setup in startup.bas or a macro file of your own naming.
- 3) If the box Use as Default? is enabled, this file will be used when Seehau is started.
- 4) This macro will save those items enabled in the Macro Save Type area.
- 5) Select No Save and exit from Seehau. This will enable a fresh start for the examples.

Figure 5



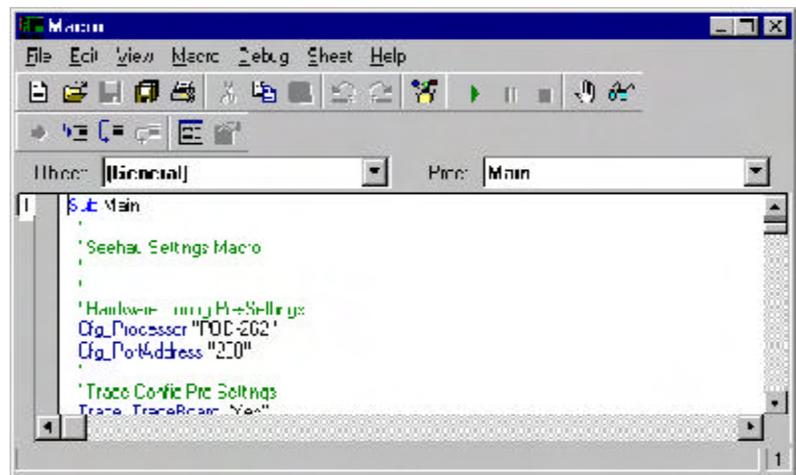
Seehau Commands

Seehau Commands are used to communicate and control the emulator hardware and the Seehau Software Debugger. These commands are used in the Seehau Macro facilities such as user macros, emulator startup configuration and for Seehau configuration items in general. Commands allow the Seehau debugger software to be a very flexible and powerful tool. Modifications and preference settings are easily made by the user. A Macro as defined here is a collection of commands in a text file that are executed by the Seehau debugger. Sax Basic is included in Seehau and permits powerful IF..Then statements to be used in your macros. If Seehau is used as an OLE Automation Server, the commands are executed from the automation controller.

Macro Construction

These Macros can be supplied by Nohau or constructed by the user using the built-in recording facilities or by hand with any ASCII editor or with the Seehau Macro Editor. The Macro Edit window is shown in Figure 6. The macro recorder and editor are found under the *Macro* menu. The resulting filename.bas files are simple text files and can be modified at any time with a text editor regardless of how the macro was originally constructed. These files are stored in the *nohau/seehau/macro* directory on your hard drive.

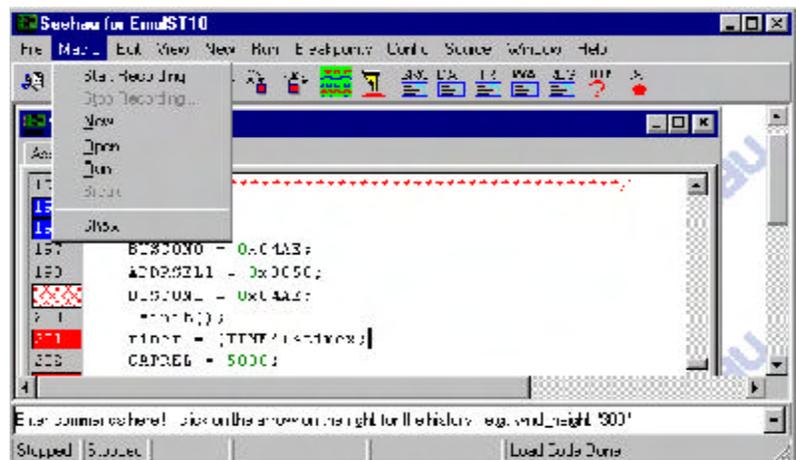
Figure 6



Macro Execution

Macros can be executed by Seehau during its startup phase or manually by the user using the *Macro/Run* command. Individual commands can be executed by direct entry in the main Seehau window. Figure 7 illustrates the dialog box at the bottom of the main window for direct entry. An example is shown. This box can be opened up to display a list of the recently used commands. This eliminates the need to retype most used commands. The box at the bottom of Figure 7 where the word "Done" is where Seehau puts the results of command operations. Not all commands display a result. Note the Macro menu item expanded at the top of the window.

Figure 7



Command Format

Commands are of the format *groupname_commandname_fields*. They are in ASCII text and the names are self-explanatory. Commands are written in upper and lower case letters for easy readability but Seehau is case insensitive here. An example of a command is: *Cfg_PortAddress "378"*. This command tells Seehau that the emulator hardware is connected to the PC port at hex 378 (LPT1). Descriptions of various fields (if applicable) are contained in the Seehau online help.

Command Examples

An example of directly entering a command:

In the dialog entry box enter: `Wnd_height "300"` and press return.

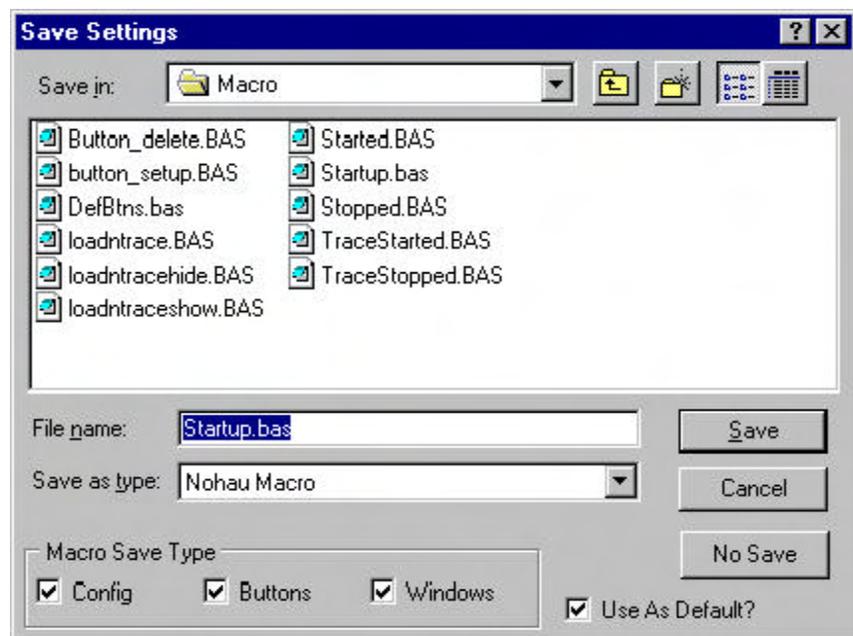
The open window or the one most recently selected by a macro command will have its height adjusted to 300 pixels. To modify the main window, you may have to type in this command first:

`Wnd_select "Seehau for Emul16x"`

This example uses the Emul 166 Emulator for the Siemens C166 family. Enter the appropriate fields for your system as indicated as the caption on the main window.

If you enter these commands in the file `/Macro/startup.bas`, they will be executed everytime you start up Seehau. You can also make your own Macro that saves Seehau configuration setup. You are given this opportunity when you leave Seehau. Figure 8 shows the options you can select:

Figure 8



Config:

This enables the saving of hardware configuration items such as breakpoints and triggers.

Buttons :

This enables the saving of the position and type of buttons displayed on the main window.

Windows :

This enables which windows are displayed and their position and size.

Use as Default ?:

The specified file will be the one used by Seehau on startup. `Startup.bas` is the initial default. If `startup.bas` is not present, Seehau will make one from a composite of various existing `.bas` files. You can specify the default to be your own Macro.

Command Groups:

Macro commands are divided into 12 groups. Each group represents one basic function. The group is specified in the first part of the command name as in *Brkpt_clrBrkPoints*. Seehau online Help files contain detailed information of all the Macro commands. Consult the online Help file for details on fields associated with the commands and for more examples.

- 1) **BrkPT:** these commands are associated with breakpoints. *i.e.* *Brkpt_clrBrkPoints* clears all the breakpoints.
- 2) **Cfg:** general emulator configuration settings. *i.e.* *Cfg_map* maps emulator memory to the target.
- 3) **Data:** operations on data such as move, search and set. *i.e.* *Data_Move* moves data from a source to a destination address.
- 4) **File:** operations on files such as symbol table and user code loads into the emulator or target memory. *i.e.* *File_LdCode filename* loads the specified user object code into the emulator.
- 5) **Hlp:** Help commands. Displays help on various subjects from the Seehau Online help file.
- 6) **Src:** Program Commands. These refer to the user code in the source window. *i.e.* *Src_SelectModule* selects the source module to be displayed in the source window.
- 7) **Reg:** CPU Register commands used to inform Seehau of the name, size and address of the CPU registers. These are used in the operation of Seehau and also to display registers in the Register window. *i.e.* *Reg_Size "16"* informs Seehau the previously selected register is 16 bits wide.
- 8) **Run:** commands used to start the emulation CPU. Commands include Run Break, Run Stepoever and Run GoForever.
- 9) **Syabr:** Used to access commands regarding the source code.
- 10) **Trace:** Commands that control the hardware Trace functions. *i.e.* *Trace_TraceBegin* is used to start the Trace.
- 11) **View:** These commands control the Inspect, Evaluate and Watch windows. *i.e.* *View_Evaluate* passes the expression to Seehau that is to be evaluated.
- 12) **Wnd:** Windows commands. These are used to control aspects of the windows including position and size, button definition and show/hide windows.

Quick Saving a Configuration Menu using the Apply Button

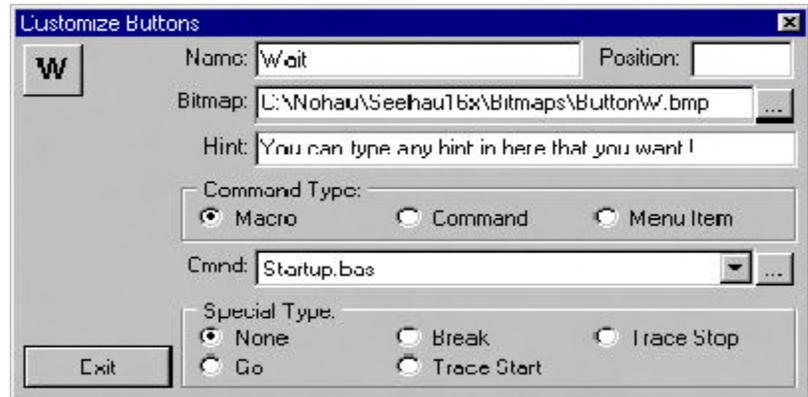
- 1) The configuration menu to be saved must be open and its Apply button must be visible.
- 2) In the Macro menu item, select Start Recording. The configuration window will disappear.
- 3) Re-open the configuration window from the appropriate menu item. Click on Apply. The settings associated with this window will be saved.
- 4) In the Macro menu, select Stop Recording.
- 5) The Save Macro window will open giving you the opportunity to choose the filename for the newly created macro. See Figure 8. Enter a filename of your choosing and click on Save. The macro is ready to use and will accurately re-create your configuration settings.

Configuration menus are also saved when general Seehau settings are saved upon exit or through the Config, Save Settings menu.

Creating New Buttons

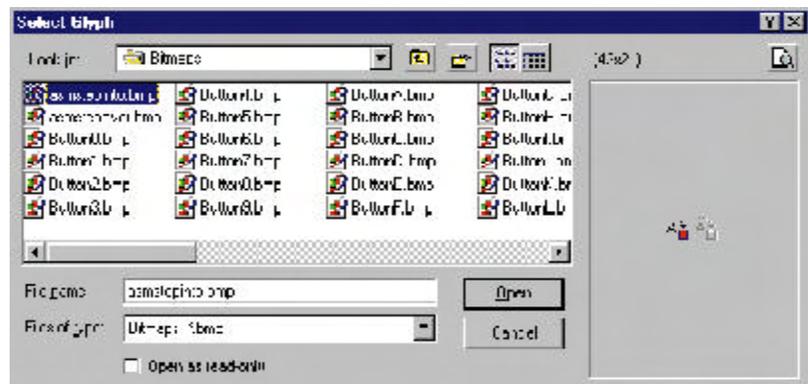
- 1) Buttons can be created and deleted on the icon bar in the main menu. A created button can be attached to a macro you have created. The icons themselves are Windows bitmaps (*.bmp) and are easily created with virtually any graphics program.
- 2) Open the menu Config, Buttons and an empty Figure 9 will open.

Figure 9



- 3) Open the bitmap browser by clicking on the ... button at the right side of the dialog box.
- 4) Figure 10 will open up showing the bitmaps in the Bitmaps directory. Note that when a bitmap is highlighted, its image appears on the right side of the window as in Figure 10.
- 5) Highlight ButtonW.bmp. Click on Open and its image will appear as in Figure 10.

Figure 10



- 6) In the Hint: dialog box shown in Figure 9, type in the sentence as shown. Position the cursor on the W icon and note this sentence will appear in a yellow box. This will be used in the main window.
- 7) In the Command Type: area, click on Macro. This area indicates what type of action will be associated with the new button. The options in the Cmd: box will change according to the Command Type setting.
- 8) In the Cmd: dialog box, select startup.bas. You can choose the down arrow to open the Macro directory or use the browser by clicking on the ... button.
- 9) Click and drag the W icon to the main window to the right of the EXIT icon. The W icon will be placed at this point. Click on Exit in the Customize Buttons window shown in Figure 9.
- 10) Confirm that if you place the cursor on the W icon the hint you entered appears.
- 11) Click on the W icon in the main window and the startup.bas sequence will be executed.
- 12) To remove a button, drag it back to the Customize Buttons window. To quickly access the Customize Buttons window, right click on the button and select Buttons. The Customize Buttons window will appear.

Chapter 4 The Nohau EMUL166: Trace and Triggers

Trace and Trigger Overview

The trace memory and triggers can be configured and viewed without intrusion into real-time emulation. This is accomplished with a dedicated 25 MHz housekeeping controller rather than stealing cycles from the emulation CPU. The triggers are versatile, yet easy to configure and modify.

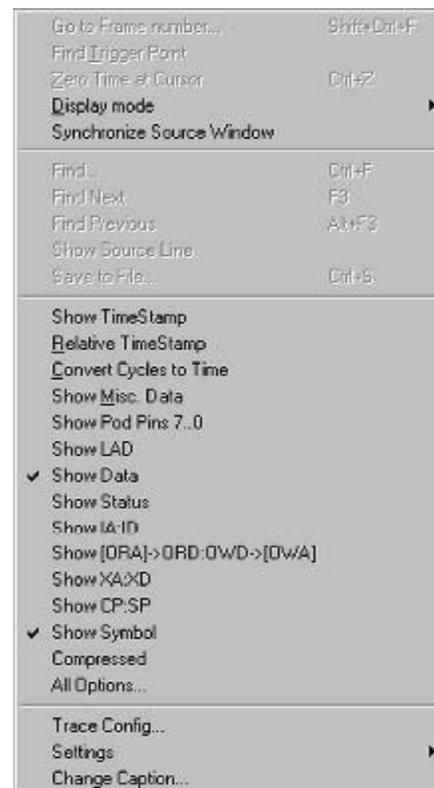
Triggers can be set on addresses and data ranges. They control trace recording or cause the emulator to stop the target depending on the options set. Trace and Triggering can trigger and record all internal and external accesses.

Full pipeline decoding ensures only executed instructions and data read/writes are captured as such and no false triggering occurs. The trace memory can be saved to a file. Source and assembly code is displayed in the trace window. The trace window can be configured as to the fields displayed.

Trace Window Display

The fields displayed in the trace window can be selected depending on your needs. Figure 1 shows the trace local window. The meaning of these fields is explained in the on-line Help files. The XDATA, ID_IA and OD_OA busses are internal CPU busses brought out by the bondout controller. These fields are available to the user. The emulator uses these busses to access the internal areas of the controller.

Figure 1



The trace recording can be manually turned off and on with the trace icon on the main menu. When the trace is not recording, the Shadow RAM also stops providing a snapshot of the systems memory.

With the timer program loaded and the trace window open, run timer for a few seconds and then stop the trace with the trace icon. This is the icon with TR indicated on it.  It will turn green.

The trace window will be filled with frames. Try activating certain features in Figure 1 to see the effects.

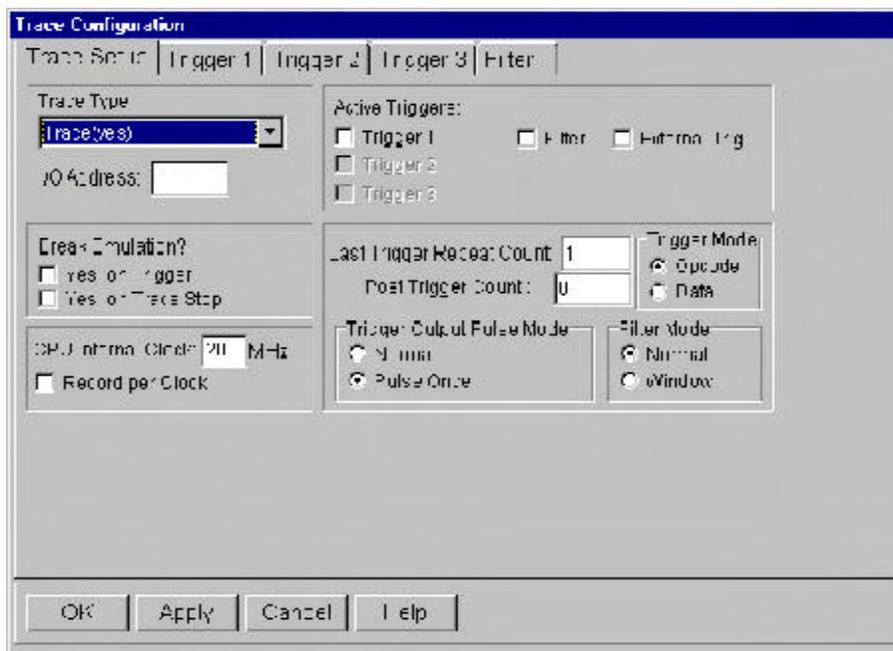
The trace memory is where the frames are stored. The triggers have the power to stop the trace recording, and/or emulation and to control what is recorded in the trace memory (filtering).

Trigger Example on an Address and Data Qualifier

Triggers are probably the most powerful facility possessed by an emulator. This example will stop the emulator when the seconds field in the clock equals 4. This is represented by a byte write to memory location 5017 with the ASCII value of 34.

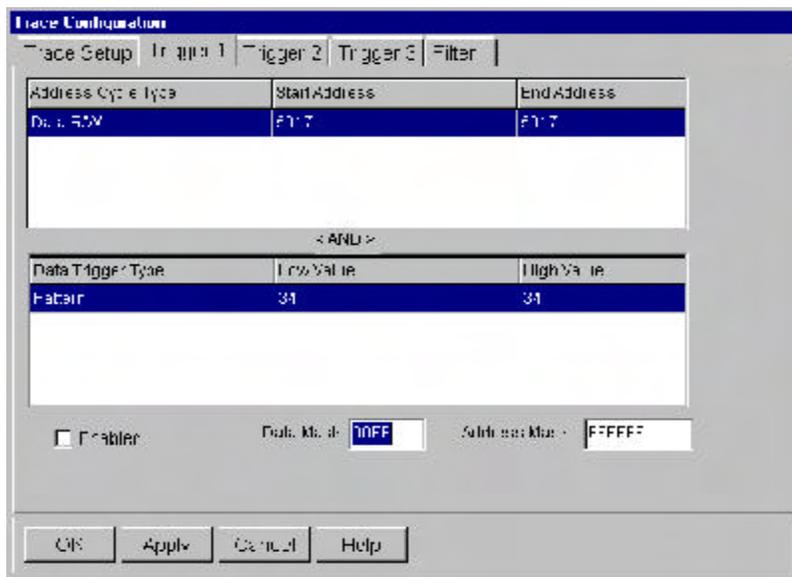
- 1) Setup the emulator to run the timer example as described in Chapter 3. Make sure the data window is set to view ASCII data as in Figure 3 in Chapter 3. This is not needed for the trigger to work but for your inspection of the 5017 memory location. Open the trace window and position the windows so the Source, Data and Trace windows are visible.
- 2) Open the menu Config, Trace from the main window. Figure 2 appears. This is where the triggers are configured. The timestamp timings are derived from the value in the CPU Internal Clock box. Note the tabs at the top showing the three Triggers and the Filter as well as the Trace Setup.

Figure 2



- 3) Activate the Yes, on the Trigger box. When a trigger occurs, the emulation will be stopped.
- 4) Click on the Trigger 1 tab. A blank Figure 3 opens up. The values of the qualifiers are inserted here.

Figure 3



- 5) Right click on the Address Cycle Type window and select Add. The Edit Trigger Qualifier dialog box in Figure 4 opens up. Fill in the fields as shown. Make sure you select Data R/W. Press OK.
- 6) Repeat for the Data Trigger Type and enter the value of 34 in each field as in Figure 5. Click OK.
- 7) In the Trace Configuration menu as shown in Figure 3, enter 00FF in the Data Mask field. Note you can use this field to enter a “don’t care” condition on a bit by bit basis, i.e. 000F will only use the lower nibble of the data qualifier. This can also be done in the address mask field to mask off address ranges.

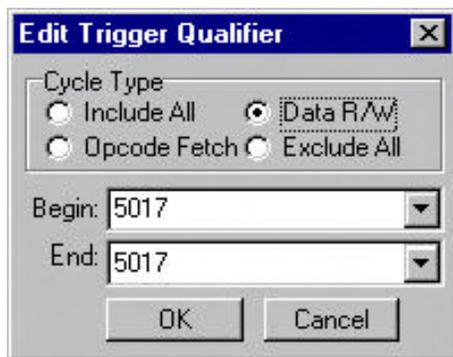


Figure 4

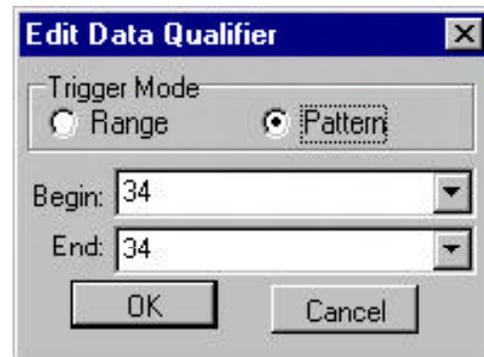


Figure 5

- 8) Click on OK to enter the data.
- 9) If the emulation and trace were running during this time, temporarily stop and start the trace to register the trigger data. Note that emulation will not be slowed or stopped during trigger setting or trace stopping.
- 10) Start emulation if not already running and when the seconds indicator reaches the value of 4, emulation will stop. The GO and TR icons will turn green and the word “Stopped” will appear twice in the lower left corner of the main menu.
- 11) You will get a trace window similar to Figure 6.

Frame	Address	Data	Opcode	Inst	Symbol
-7	470:	F207	07F23C00	ADDB	R11, #30h
-6	472:	J03J	0030 -- 002		
-5	474:	3C8D	B03C	MOVW	[R12], R11 ← A
-4	476:	C10B	08C1	ADD	R12, #1h
-3	478:	F2E7	E7F23A00	MOVW	R11, #31h
-2	47A:	003A	003A -- 002		
-1	5017:	34	34 -- w1		← B
0	47C:	2C89	B92C	MOVW	[R12], R11
1	47E:	00CD	00CD	RET	
2	480:	9089	88E0	MOV	[-R0], R9 ==> CHECK_TIMER!

Frames: 3: 068(-310682), Trig Count: 1

Figure 6

- 12) The arrow at B is the write of 34 to address 5017. w1 means a 1 byte write bus cycle. The instruction that did write this is at arrow A.
- 13) The instructions at Frame -4 and -3 are prefetches to the CPU pipeline. The fact that the instruction fetch at A is followed later by the write at B clearly shows the pipeline effect. Frame -2 is the subsequent word fetch corresponding to the first fetch shown in Frame -3. oo2 signifies a 2 byte fetch or a 2 byte pipeline flush. Frame -2 is repeated inside Frame -3 where it was disassembled. Frame -2 is for reference to the bus cycle that actually occurred. It is placed in Frame -2 for readability.

Trace Filter Example

This example uses the Filter tab to record only certain frames in the trace memory. We will record only byte writes of 34 to address 5017 in the trace memory.

- 1) Open the Trace Configuration menu in the Config menu item. Figure 2 will open.
- 2) Deselect the Trigger 1 checkbox. This disables the settings we placed in Trigger 1.
- 3) Deselect the Break on Emulation? checkbox.
- 4) Click on the Filter tab.
- 5) Enter the data in the same fashion as in Figures 3, 4 and 5. Except this time in the Filter tab menu.
- 6) Set the Data Mask to 00FF as before.
- 7) Click on OK to enter the data.
- 8) Start the emulation as before and let it run for a few seconds and stop emulation.
- 9) Figure 7 will be displayed. Note that only byte writes of 34 to location 5017 are recorded.

Frame	Address	Relative time	Opcode	Inst	Symbol
11	5017:	67.375 us	36	w1	
-10	5017:	67.375 us	36	-- w1	
-9	5017:	67.375 us	36	-- w1	
8	5017:	67.375 us	36	w1	
-7	5017:	67.375 us	36	-- w1	
-6	5017:	67.375 us	36	-- w1	
5	5017:	67.375 us	36	w1	
-4	5017:	67.375 us	36	-- w1	
-3	5017:	67.375 us	36	-- w1	
-2	5017:	67.375 us	36	-- w1	

Frames: 101071(101071:1), Trig Count: 0

Figure 7

Note the timestamp. It shows that each write was 67 usec apart. There are many other combinations of triggers and filters you can use. In each of the Address Cycle and Data Trigger Type fields you can enter 50 qualifiers.

This filter mode has been set to the Normal mode. This mode records qualifiers within a range of addresses OR data values. If you set this to record a function, the functions called by this function will not be recorded.

With the Window mode, Trigger 1 is used to start the trace recording and Trigger 2 is used to stop the recording. The called functions from within a specified function will be recorded in this case. To select start and stop filtering, click on the window check box under Filter Mode in Figure 2.

Chapter 5 The Nohau EMUL166: Sabre Trace and Triggers

Trace and Trigger Overview

The trace memory and triggers can be configured and viewed without intrusion into real-time emulation. This is accomplished with a dedicated 25 MHz housekeeping controller rather than stealing cycles from the emulation CPU. The triggers are versatile, yet easy to configure and modify.

Triggers can be set on addresses and data ranges. They control trace recording or cause the emulator to stop the target depending on the options set. Trace and Triggering can trigger and record all internal and external accesses.

Full pipeline decoding ensures only executed instructions and data read/writes are captured as such and no false triggering occurs. The trace memory can be saved to a file. Source and assembly code is displayed in the trace window. The trace window can be configured as to the fields displayed.

Trace Window Display

The fields displayed in the trace window can be selected depending on your needs. Figure 1 shows the trace local window. The meaning of these fields is explained in the on-line Help files. The Internal ROM Bus, Operand Read/Write and CP:SP busses are internal CPU busses brought out by the bondout controller. These fields are available to the user. The emulator uses these busses to access the internal areas of the controller.

Figure 1



The trace recording can be manually turned off and on with the trace icon on the main menu. When the trace is not recording, the Shadow RAM also stops providing a snapshot of the systems memory.

With the timer program loaded and the trace window open, run timer for a few seconds and then stop the trace with the trace icon. This is the icon with TR indicated on it.  It will turn green.

The trace window will be filled with frames. Try activating certain features in Figure 1 to see the effects.

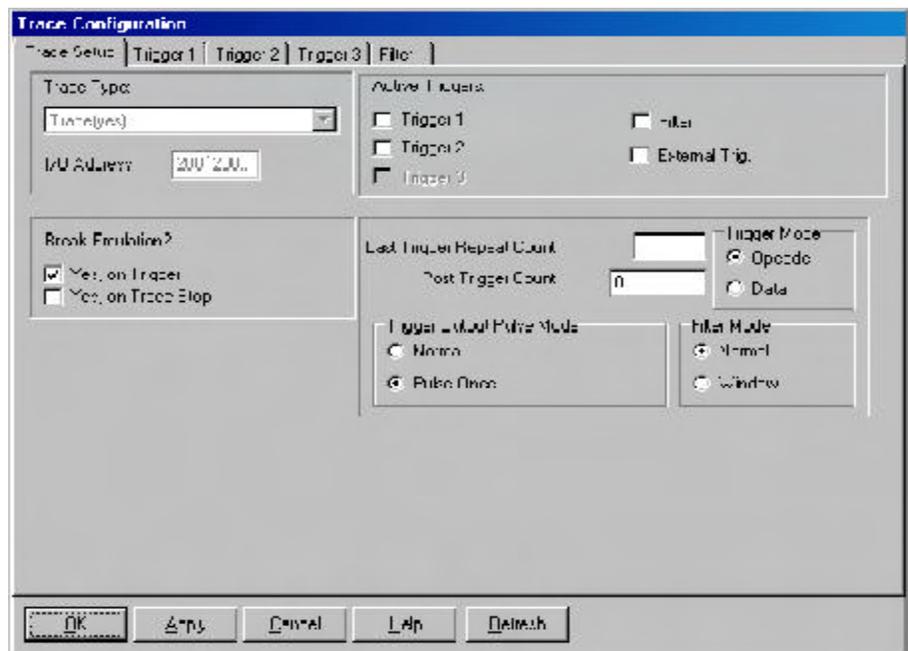
The trace memory is where the frames are stored. The triggers have the power to stop the trace recording, and/or emulation and to control what is recorded in the trace memory (filtering).

Trigger Example on an Address and Data Qualifier

Triggers are probably the most powerful facility possessed by an emulator. This example will stop the emulator when the seconds field in the clock is greater than 30. This is represented by a byte write to memory location 5016 with the ASCII value of 33.

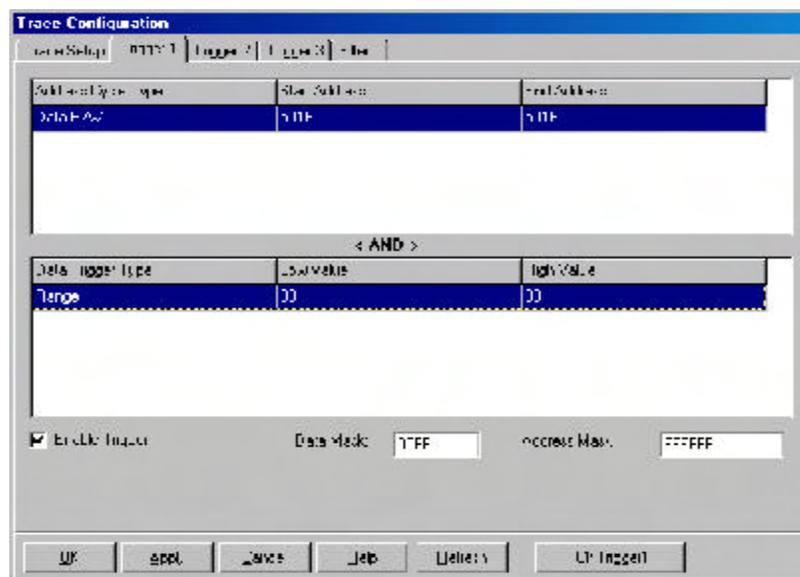
- 1) Setup the emulator to run the timer example as described in Chapter 3. Make sure the data window is set to view ASCII data as in Figure 3 in Chapter 3. This is not needed for the trigger to work but for your inspection of the 5016 memory location. Open the trace window and position the windows so the Source, Data and Trace windows are visible.
- 2) Open the menu Config, Trace from the main window. Figure 2 appears. This is where the triggers are configured. The timestamp timings are derived from the value in the CPU Internal Clock box. Note the tabs at the top showing the three Triggers and the Filter as well as the Trace Setup.

Figure 2



- 3) Activate the Yes, on the Trigger box. When a trigger occurs, the emulation will be stopped.
- 4) Click on the Trigger 1 tab. A blank Figure 3 opens up. The values of the qualifiers are inserted here.

Figure 3



- 5) Right click on the Address Cycle Type window and select Add. The Edit Trigger Qualifier dialog box in Figure 4 opens up. Fill in the fields as shown. Make sure you select Data R/W. Press OK.
- 6) Repeat for the Data Trigger Type and enter the value of 34 in each field as in Figure 5. Click OK.
- 7) In the Trace Configuration menu as shown in Figure 3, enter 00FF in the Data Mask field. Note you can use this field to enter a “don’t care” condition on a bit by bit basis, i.e. 000F will only use the lower nibble of the data qualifier. This can also be done in the address mask field to mask off address ranges.

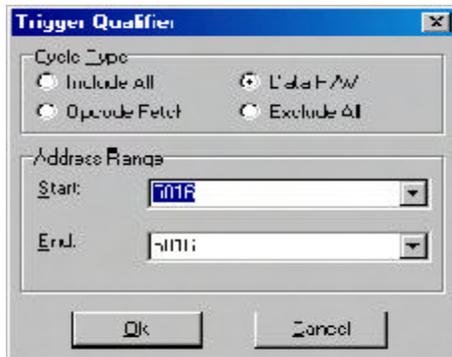


Figure 4

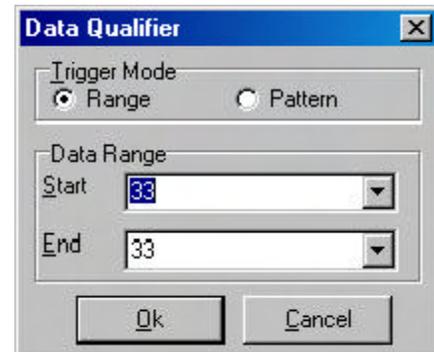


Figure 5

- 8) Click on OK to enter the data.
- 9) If the emulation and trace were running during this time, temporarily stop and start the trace to register the trigger data. Note that emulation will not be slowed or stopped during trigger setting or trace stopping.
- 10) Start emulation if not already running and when the seconds indicator reaches the value of 4, emulation will stop. The GO and TR icons will turn green and the word “Stopped” will appear twice in the lower left corner of the main menu.
- 11) You will get a trace window similar to Figure 6.

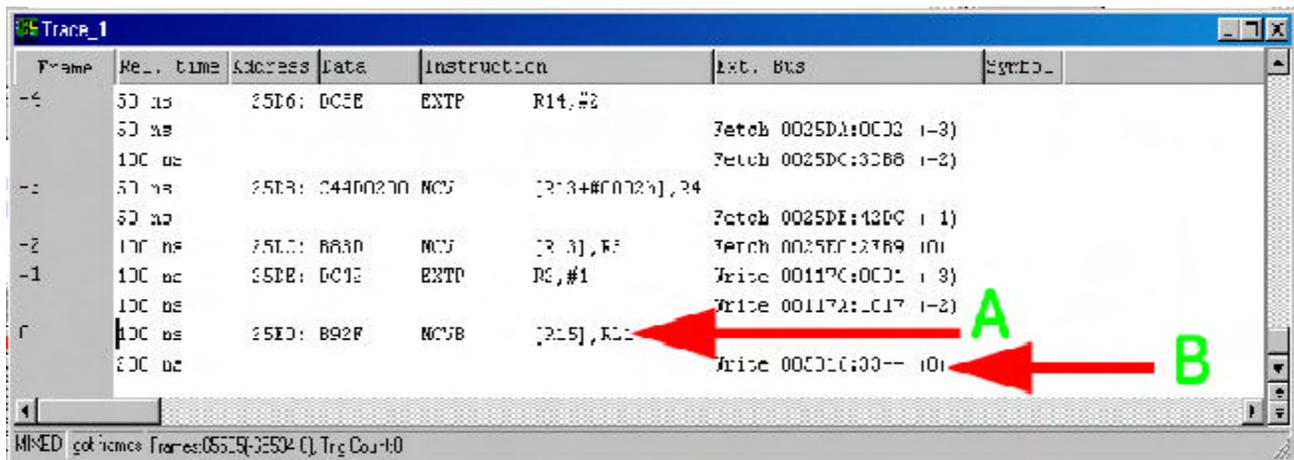


Figure 6

- 12) The arrow at B is the write of 33 to address 5016. The single value after the address means a 1 byte write bus cycle. The instruction that did write this is at arrow A.

Trace Filter Example

This example uses the Filter tab to record only certain frames in the trace memory. We will record only byte writes of 34 to address 5017 in the trace memory.

- 1) Open the Trace Configuration menu in the Config menu item. Figure 2 will open.
- 2) Deselect the Trigger 1 checkbox. This disables the settings we placed in Trigger 1.
- 3) Deselect the Break on Emulation? checkbox.
- 4) Click on the Filter tab.
- 5) Enter the data in the same fashion as in Figures 3, 4 and 5. Except this time in the Filter tab menu.
- 6) Set the Data Mask to 00FF as before.
- 7) Click on OK to enter the data.
- 8) Start the emulation as before and let it run for a few seconds and stop emulation.
- 9) Figure 7 will be displayed. Note that only byte writes of 33 to location 5016 are recorded.

Frame	Rel. time	Address	Data	Instruction	Expr. Tms	Symbol
-4	207 ns	22E0: B92F	Y0WF	[R.F],R1		
	203 ns				Write 0C5015:38	1-31
	404.450 us				Fetch 0C25E0:37B9	1-31
-0	303 ns	22E0: B92F	Y0WE	[R.F],R1		
	207 ns				Write 0C5015:30	1-31
	404.750 us				Fetch 0C25E0:37B9	1-31
2	207 ns	22E0: B92F	Y0WT	[R.F],R1		
	203 ns				Write 0C5015:38	1-31
	404.450 us				Fetch 0C25E0:37B9	1-31
-1	303 ns	22E0: B92F	Y0WE	[R.F],R1		
	203 ns				Write 0C5015:30	1-31

Figure 7

Note the timestamp. It shows that each write was 400 nsec apart. There are many other combinations of triggers and filters you can use. In each of the Address Cycle and Data Trigger Type fields you can enter 50 qualifiers.

This filter mode has been set to the Normal mode. This mode records qualifiers within a range of addresses OR data values. If you set this to record a function, the functions called by this function will not be recorded.

With the Window mode, Trigger 1 is used to start the trace recording and Trigger 2 is used to stop the recording. The called functions from within a specified function will be recorded in this case. To select start and stop filtering, click on the window check box under Filter Mode in Figure 2.

Chapter 6 The Nohau EMUL166: Connecting to your Target

Clocks, Oscillators and Crystals

The maximum frequency of the EMUL166-E3 is 40 MHz. This is the ECLK signal located on the emulator board. The most used frequency is 20 MHz. This is usually supplied by a 5 MHz crystal or oscillator which is then multiplied by 4 to obtain the required 20 MHz. See the Siemens User manual for more information on clocks.

Crystal Capacitors

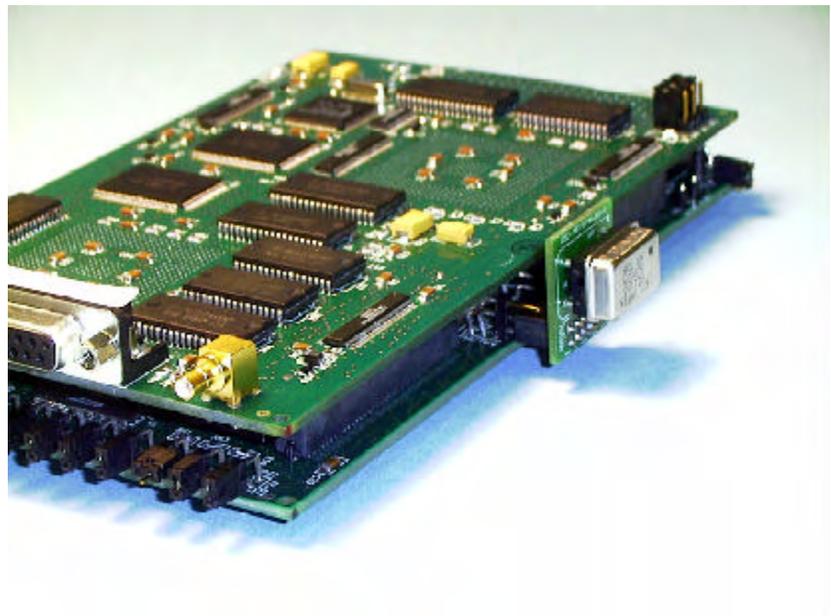
If your application uses a crystal please note that Siemens specifies the capacitors are two different values. While most crystal oscillators use capacitors of the same value, stability is increased with different values. Siemens have an application note on this subject named AP242003 and it can be found on the Infineon web at <http://www.infineon.com/products/ics/34/index3.htm>. Note that many Phytex evaluation boards use same value capacitors and you may need to add an additional capacitor to ensure adherence to the Siemens specification. Suggested values at 5 MHz are 22 pf (input) and 47 pf (output) with a resistor value of 390 ohms.

EMUL166 Oscillators

Earlier models of Nohau emulators had a plug in oscillator package of either 20 or 5 MHz. Later models have this oscillator soldered in for reliability. Nohau offers a simple adapter board so you can easily change oscillator values. This is shown in Figure 1 and both oscillator and crystal versions are offered.

This adapter connects to JP51, JP7 and JP10. The adapter will plug into the lower set of jumpers i.e. target. Some versions may have a PLL IC and the ability to send the clock to the target. Contact Nohau Technical Support for the latest information. support@icetech.com

Figure 1



Connecting to Your Target Hardware

Connecting an emulator to a target system can require some careful work. The Nohau EMUL166 mimics the C166 family as closely as possible. There are a few issues that can cause particular problems. Here are some useful hints:

- 1) Try the emulator in stand-alone mode first. Get it to work this way first. You do not have the complications of your target buses to make things harder to resolve.
- 2) If you connect the emulator to the target without making any changes to your jumper or software settings: the emulator should still work. If it does not, check for shorted or incorrect signal lines on the target. Get someone else to check your board layout. Many problems are finally traced to crossed lines.
- 3) Once you have the emulator working from its internal memory - you can switch to the target memory.
- 4) Remove the RESET and READY jumpers. If your reset stays on too long, the emulator will never run.
- 5) Measure the CPU frequency at the ECLK pad on the emulator. See Figure 3 in Chapter 1. Make sure it is what you expect it to be. If your program crashes, make sure the frequency is not changing unexpectedly.
- 6) Most users prefer to bring the target's clock up to the bondout controller. Make sure your clock pins are correct. JP10 and JP7 should be in the lower position.
- 7) If you are supplying power to the target and the emulator separately, make sure JP6 T-PWR is not connected. Make sure you use the proper power sequence to protect the bondout. Never allow the target to have power without the emulator being powered up.
- 8) Consider purchasing an evaluation board to serve as a reference design. Phytex boards are good choices but you should consider that some do not use the Siemens chip selects but rather a GAL device for addressing. The C161 and C164 are like this. The early C161 does not use the Siemens bootstrap mode. Adapters are very inexpensive for these boards. The C167 board uses CS0 for the FLASH and CS1 for the RAM and this setup is very easy to use.
- 9) To test target RAM, use the data window to change a memory location to different values such as FF or 00. The emulator must return the same value that you entered. You can use any erroneous bit patterns to help determine where the error is.
- 10) Design small test programs that you can send to Nohau technical support. Please include the source and any compiler project files if these are used. The ability to replicate the problem is important and helps a great deal.
- 11) The reset location (00 0000) will normally contain a valid jump instruction after the user code has been loaded. If it does not, make sure you are in the correct address space such as external or single-chip mode and that you are not accessing some ROM.
- 12) If you have some RAM on your target and you have this mapped to your emulator: you should be able to write values to this RAM from within a Data window and get the same value returned. You should be able to do this on adjacent bytes and also on a whole word. If you do not get the same value returned, you are either in some sort of ROM, nonexistent or defective RAM or a setting could be incorrect.

If you can write correctly to one byte yet not to the next, this nearly always means adapter problems or crossed data lines on your target..

Conclusions

This explains a portion of the power of Seehau and Nohau C166 family emulators. We have not explored the RTOS support, code coverage, compiling and editing abilities of Seehau.

For more information, contact your local Nohau representative or Nohau at www.icetech.com.

For applicable Application Notes see www.icetech.com under Technical Publications.